ERDC MSRC PET Technical Report No. 01-24

Adaptive Mesh Refinement in CTH: Implementation of Block-adaptive Multi-material Refinement and Advection Algorithms

by

David L. Littlefield
J. Tinsley Oden
Graham F. Carey

15 May 2001

Work funded by the Department of Defense
High Performance Computing Modernization Program
U.S. Army Engineer Research and Development Center
Major Shared Resource Center through

Programming Environment and Training

Supported by Contract Number: DAHC94-96-C0002
Computer Sciences Corporation

Adaptive Mesh Refinement in CTH: Implementation of Block-adaptive Multi-material Refinement and Advection Algorithms

David L. Littlefield, J. Tinsley Oden and Graham F. Carey
Texas Institute for Computational and Applied Mathematics
The University of Texas at Austin
Austin, TX 78712

e-mail: littlefield@ticam.utexas.edu

ABSTRACT

An adaptive mesh refinement (AMR) version of CTH is currently under development. This project is being conducted jointly by researchers at the University of Texas and at Sandia National Laboratories. The AMR version of CTH represents a significant milestone in the ten-year development of this legacy code.

CTH is a multi-material wave propagation code used by many analysts in the DoD user community to simulate large deformations, large strain rates, and strong shocks in solid, liquids and gases. The numerical procedure is based on a finite volume formulation of very general forms of the continuum equations. As such, it can be applied to a wide variety of problems. The computational mesh used in CTH is Eulerian; materials and material interfaces are permitted to flow through the mesh as the calculation proceeds.

The incorporation of new algorithms into the AMR version of CTH is described. A block refinement algorithm that preserves the location of material interfaces has been implemented into a working version of the code. This algorithm uses advanced interface tracking to map materials; this minimizes the dispersion normally associated with the process. A multi-material advection algorithm, which is basically a generalization of Youngs' interface reconstruction method to cells with mismatched faces, is described. Results from three-dimensional examples problems are shown that effectively illustrates the improvements afforded by these new algorithms.

ADAPTIVE STRATEGY

When implementing adaptive refinement into an existing code, it is very important to consider the organization and data structure of the target application code. Here, the application code is CTH (McGlaun *et al.* 1990), a three-dimensional multi-material Eulerian wave propagation code designed for modeling very large deformations and strong shocks. The data in CTH is organized in *(I,J,K)* logical blocks that correspond to the mesh used in the problem. Within a block, the mesh contours are constrained to remain parallel with the coordinate axes, and the introduction of *hanging*, or constrained, nodes is not permitted. However, adjacent blocks are permitted to have different values of *I, J* and *K*. Thus, a reasonable approach for the implementation of adaptivity, which preserves the original data structure used in CTH, is to limit refinement/unrefinement to the block level. Furthermore, in order to simplify the algorithms for communication between blocks, the refinement/unrefinement was limited to isotropic 2:1 ratios between adjacent blocks. This process is illustrated in Fig. 1, where a set of communicating blocks is shown. The contents of the ghost cells along the periphery of the blocks are provided by information coming from adjacent blocks. Since these adjacent blocks may be at a different resolution, calculating the contents of the ghost cells may involve a cell split/combine process.

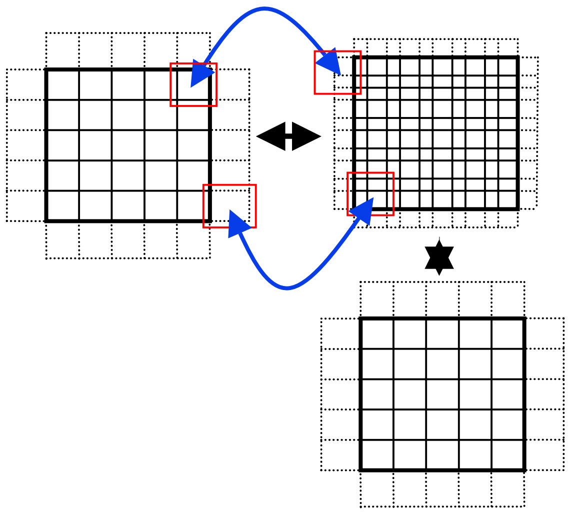

Figure 1. Block-adaptive strategy applied to target application code.

A significant part of this effort was to establish the two-way communication between blocks, as well as to make the scheme work in parallel, which is the subject of another paper (Crawford 1999). The focus of this work is on the development of refinement schemes, as well as error indicators, suitable for implementation into a three-dimensional Eulerian shock physics code.

REFINEMENT/UNREFINEMENT

The collapse of 8 child cells (in three dimensions) into a single parent cell is a simple process, which for brevity will not be completely described here. For example, intensive quantities (such as specific internal energy) in the parent cell are mass averages of the intensive quantities in the child cells, masses and volumes are simply sums of the values from the child cells, and material volume fractions are volume averages from the child cells.

The refinement process, on the other hand, requires a parent cell to be split into 8 child cells. This process is complicated by the fact that material interfaces exist within the cell; thus the location of these interfaces must be preserved when the refinement is done. To accomplish this, it is useful to review the algorithms used for interface tracking in CTH.

Review of Interface Tracking

In CTH, the location of material interfaces within a cell is interpreted using Youngs' algorithm (Youngs 1987). Youngs' algorithm is basically a systematic procedure for determining the position and orientation of the interface plane separating two materials in a computational cell, given the volume fractions of materials in the cell as well as the surrounding cells.

The basic procedure used in Youngs' algorithm is to determine the outward unit normal vector **n** and the perpendicular distance d from the interface plane to a reference corner. These two quantities uniquely determine the position and orientation of the interface plane. The normal **n** is determined readily from the volume fractions as

$$\mathbf{n} = -\frac{\nabla\phi}{\left|\nabla\phi\right|}\,, \qquad (1)$$

where ϕ is the volume fraction of the material of interest. Likewise, d is determined based on its value from one of five possible intersection conditions, given in Fig. 2. These include the triangle section, two quadrilateral sections, a pentagonal section, and a hexagonal section. Formulas for d in each of these configurations, as well as the values for **n** and ϕ where each of these is applicable, can be found elsewhere (Youngs 1987).

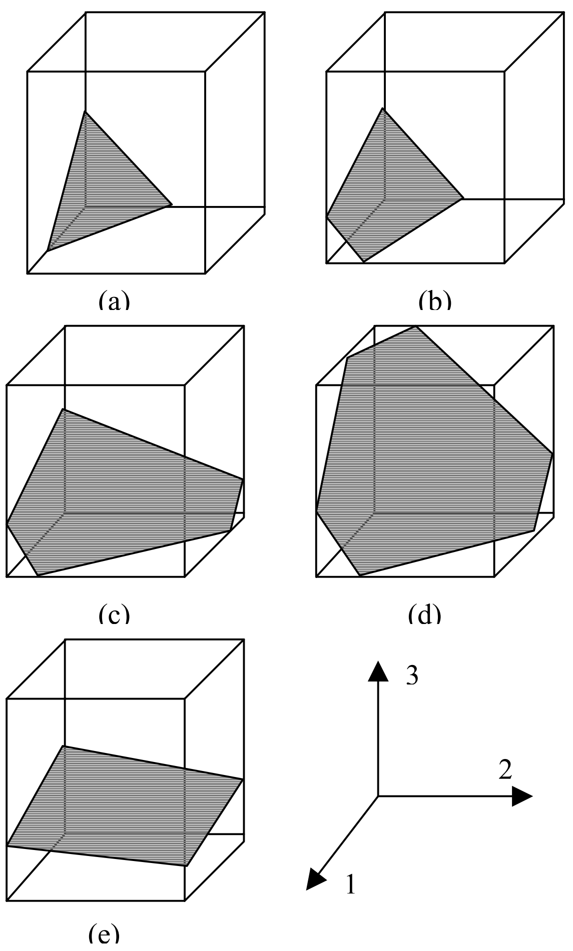

Figure 2. Possible intersection conditions for a plane intersecting a unit cube: (a) triangle section, (b) quadrilateral section A, (c) pentagonal section, (d) hexagonal section, and (e) quadrilateral section B.

Extension to Cell Refinement

Youngs' algorithm can also be extended to insure a consistent interface mapping for cell refinement. Once **n** and d for a parent cell are known, values for **n** and d can be recovered for the eight child cells in a manner that preserves the interface mapping of the original parent cell. For example, consider the refinement of a parent cell into 8 child cells as is depicted in Fig. 3. The unit normal vector

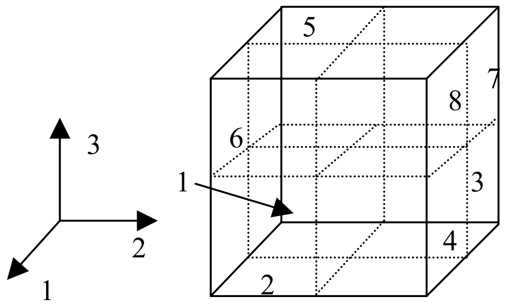

Figure 3. Schematic of refinement of a parent cell into eight equal volume child cells.

n is the same for the eight children as it is for the parent. The values for d, on the other hand, are given by

$$\begin{aligned}
d_1 &= 2d\ , \\
d_2 &= 2d - n_1\ , \\
d_3 &= 2d - n_2\ , \\
d_4 &= 2d - (n_1 + n_2), \\
d_5 &= 2d - n_3\ , \\
d_6 &= 2d - (n_1 + n_3), \\
d_7 &= 2d - (n_2 + n_3), \\
d_8 &= 2d - (n_1 + n_2 + n_3),
\end{aligned} \tag{2}$$

where d_i denotes the value of d for the child cells depicted in Fig. 3. Note that the formulas for d_i given in Eq. (2) are values corresponding to a unit cell.

Once the values for **n** and d_i in each of the child cells are known, a procedure can be followed for determining the volume fractions for materials in each of the children. The number of intersection conditions that must be considered again reduces to the five possibilities given in

Fig. 2. Formulas for ϕ in each of these configurations, as well as ranges for **n** and d_i where each of these is applicable, can be found elsewhere (Littlefield and Oden 1999).

BLOCK-ADAPTIVE MULTI-MATERIAL ADVECTION

The location of material interfaces within a cell also has an effect on the advection of mass, momentum and energy to adjacent cells. As such, in incorporating a block-adaptive scheme, it is important to consider the details of material interfaces on the advection between blocks of different mesh resolutions.

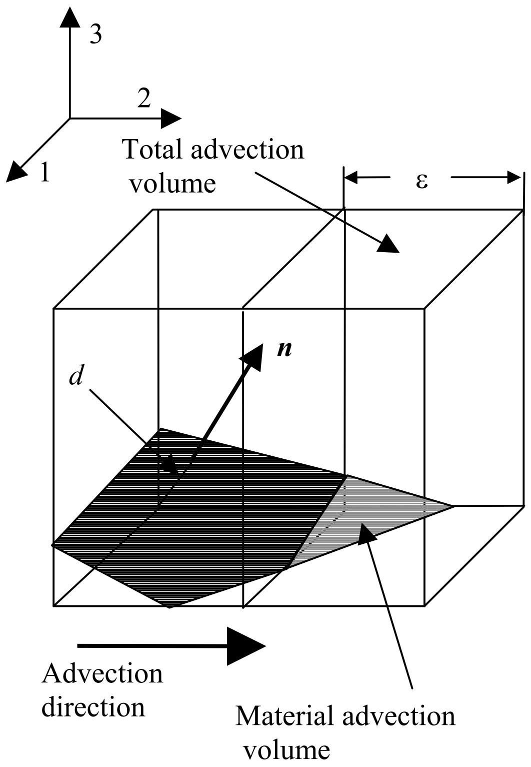

Figure 4. Illustration of multi-material advection.

Review of Multi-Material Advection

In order to understand the procedure for implementation of block-adaptive multi-material advection, it is instructive to review the standard multi-material algorithm in CTH. A second-order algorithm is used to advect variables between adjacent cells. Central to the algorithm is a determination of the volume fractions of materials in the advection volume. A variant of Youngs' algorithm is used for this purpose.

Consider advection occurring to the right from a unit cube, as is illustrated in Fig. 4. The total advection volume is denoted as ε. The material advection volume is the volume enclosed by the intersection of the interface plane with the total advection volume. Youngs' original method outlines an extensive set of formulas needed to determine the volume fraction of the material enclosed in the total advection volume. However, a much simpler method is to simply re-normalize the values for **n** and d for the interface plane with respect to the total advection volume. This technique is used in a number of Eulerian hydrocodes (including CTH), but to the author's knowledge has never been published in the open literature. For example, for advection occurring in the +2 direction, the proper normalizations are:

$$\begin{aligned}
\tilde{n}_1 &= n_1\left[n_1^{\,2} + (\varepsilon n_2)^2 + n_3^2\right]^{-1/2} ,\\
\tilde{n}_2 &= \varepsilon n_2\left[n_1^{\,2} + (\varepsilon n_2)^2 + n_3^2\right]^{-1/2} ,\\
\tilde{n}_3 &= n_3\left[n_1^{\,2} + (\varepsilon n_2)^2 + n_3^2\right]^{-1/2} ,\\
\tilde{d} &= \left[d + n_2(\varepsilon - 1)\right]\left[n_1^{\,2} + (\varepsilon n_2)^2 + n_3^2\right]^{-1/2} ,
\end{aligned} \qquad (3)$$

where ~ denotes the appropriate quantity normalized to the advection volume. Note that the value for **ñ** for the material interface in the advection volume is the same value as for the cell volume, but the value for $\tilde{d}$ is translated so that it is applicable to the interface in the total advection volume, *not* the cell volume. Here, it is understood that a negative value for $\tilde{d}$ implies that the material interface does not intersect the advection volume.

Once the values for $\tilde{\mathbf{n}}$ and $\tilde{d}$ have been determined, the same set of formulas used to determine material volume fractions in the refined cells (e.g., see Littlefield and Oden 1999) can be used again to determine the volume fractions of materials in the advection volume.

Extension to Block-Adaptive Multi-Material Advection

This technique can also be extended to determine the materials contained in advection volumes across mesh blocks that are at different resolutions. For example, consider four adjacent cells (in three dimensions), each one of these similar to that shown in Fig. 4, advecting to a low-resolution cell lying to the right. In that case, the volumes of materials advected to the low-resolution cell are simply the sums of the material volumes coming from the high-resolution cells. However, when a low-resolution cell advects to four high-resolution cells, the split of materials to the cells must be accomplished in a manner that preserves interfaces. Consider the division of an advection volume into four high-resolution advection

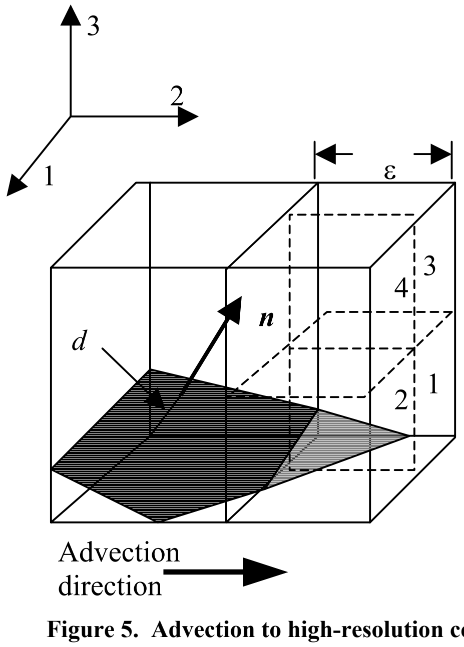

Figure 5. Advection to high-resolution cells.

volumes, as shown in Fig. 5. The normalization given in Eq. (3) determines the values for **n** and d in the advection volume. A second normalization determines the values for **n** and d in the four high-resolution cells. For example, for advection in the +2 direction the components of the unit normal become:

$$\begin{aligned}
\bar{n}_1 &= \tilde{n}_1/2\left[(\tilde{n}_1/2)^2 + \tilde{n}_2{}^2 + (\tilde{n}_3/2)^2\right]^{-1/2}, \\
\bar{n}_2 &= \tilde{n}_2\left[(\tilde{n}_1/2)^2 + \tilde{n}_2{}^2 + (\tilde{n}_3/2)^2\right]^{-1/2}, \\
\bar{n}_3 &= \tilde{n}_3/2\left[(\tilde{n}_1/2)^2 + \tilde{n}_2{}^2 + (\tilde{n}_3/2)^2\right]^{-1/2},
\end{aligned} \tag{4}$$

where – denotes components in the high-resolution cells. These normals are the same for each of the four cells. Likewise, the values for d become:

$$\begin{aligned}
\bar{d}_1 &= \tilde{d}\left[(\tilde{n}_1/2)^2 + \tilde{n}_2^2 + (\tilde{n}_3/2)^2\right]^{-1/2}, \\
\bar{d}_2 &= \left(\tilde{d} - \tilde{n}_1/2\right)\times \\
&\qquad \left[(\tilde{n}_1/2)^2 + \tilde{n}_2^2 + (\tilde{n}_3/2)^2\right]^{-1/2}, \\
\bar{d}_3 &= \left(\tilde{d} - \tilde{n}_3/2\right)\times \\
&\qquad \left[(\tilde{n}_1/2)^2 + \tilde{n}_2^2 + (\tilde{n}_3/2)^2\right]^{-1/2}, \\
\bar{d}_4 &= \left(\tilde{d} - \tilde{n}_1/2 - \tilde{n}_3/2\right)\times \\
&\qquad \left[(\tilde{n}_1/2)^2 + \tilde{n}_2^2 + (\tilde{n}_3/2)^2\right]^{-1/2},
\end{aligned} \tag{5}$$

where $\bar{d}_i$ denotes the value of $\bar{d}$ for each of the high-resolution advection volumes depicted in Fig. 5. Note that a negative value for any of the d_i's implies that this interface does not intersect the cell.

With the values for $\bar{\mathbf{n}}$ and $\bar{d}$ known, the material volume fractions for each of the high-resolution advection volumes are determined using the same formulas for cell refinement referred to previously (Littlefield and Oden, 1999).

ERROR ESTIMATION

A novel formulation for estimates of the error in the calculation, based on estimates of cell residuals, was developed as a means for triggering refinement and unrefinement. This formulation has been presented elsewhere (Littlefield and Oden 1999), and for brevity will not be repeated here.

PROGRESS AND CALCULATIONS

Integration of the algorithms described herein is work still in progress; the high-resolution refinement algorithm has been implemented into CTH, but the advection algorithm is not yet complete. Further, the residual error estimates for triggering refinement/unrefinement have not yet been implemented. Nevertheless, even without these algorithms, preliminary results are promising and effectively illustrate the benefits of adaptivity to these types of computations.

Figs. 6 – 8 show a sequence of images from a typical simulation. In this calculation, an aluminum sphere impacts an aluminum plate at 5 km/s. The initial velocity vector subtends an angle of 45° with respect to the normal of the plate, and the ratio of the sphere diameter to the plate thickness is 1.5. The initial impact configuration is shown in Fig. 6, where materials and mesh blocks are shown (each block represents a 12x12x12 uniform mesh in the calculation). By 2.6 μs, Fig. 7 shows that the sphere and the plate are severely deformed and bulged from the impact. Block refinement in the bulged region behind the plate is apparent. Fig. 8 shows fragments of the plate and sphere propagating across the mesh; block refinement in the regions containing these fragments is evident.

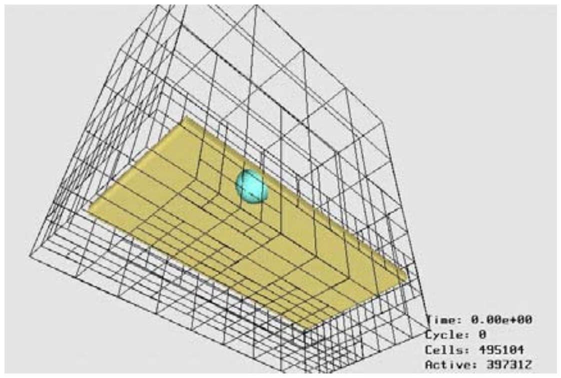

Figure 6. Impact of an aluminum sphere on an aluminum plate - initial impact geometry.

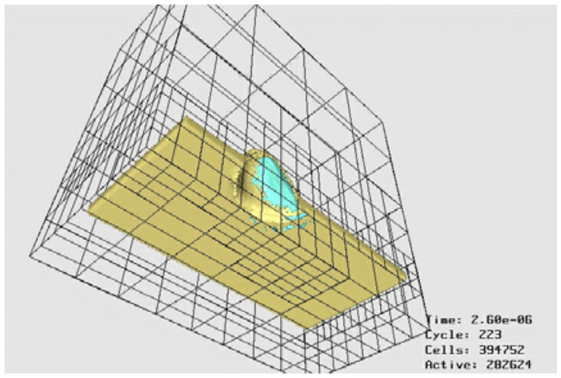

Figure 7. Impact of an aluminum sphere on an aluminum plate – 2.6 μs after impact.

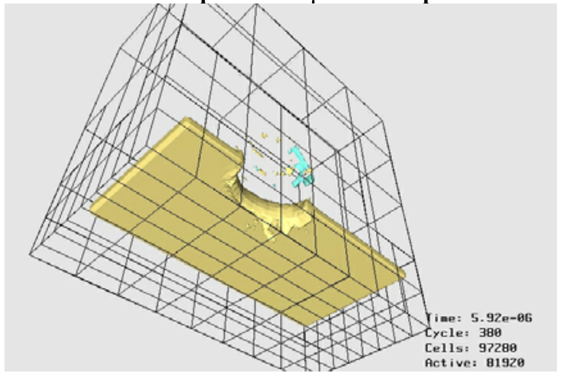

Figure 8. Impact of an aluminum sphere on an aluminum plate – 5.9 μs after impact.

CONCLUSIONS

A selection of algorithms necessary for integration of block-adaptive mesh refinement in an Eulerian impact mechanics code is described. Complete integration of these algorithms is in progress. Preliminary results demonstrate that adaptive mesh refinement can significantly improve the computational efficiency of this class of simulations. The implications for high-performance computing applications are profound: since these problems typically push the memory and CPU limits of *any* computing platform, improvements in computational efficiency with adaptive mesh refinement increases the size and/or decreases the resources required for simulation.

ACKNOWLEDGEMENTS
This work was supported by the DoD High Performance Computing Modernization Program EDRC Major Shared Resource Center through Programming Environment and Training (PET). The authors would also like to acknowledge Dr. Rick Weed for his suggestions and comments.

REFERENCES

Crawford, D. 1999. "Adaptive mesh refinement in CTH", *Proceedings of the U. S. Army Symposium on Solid Mechanics*, Myrtle Beach SC, April 11-14.

Littlefield, D. L. and Oden, J. T. 1999. "Implementation of Adaptive Mesh Refinement in an Eulerian Hydrocode", ERDC MSRC PET Preprint.

McGlaun, J. M., Thompson, S. L. and Elrick, M. G. 1990. "CTH: A three dimensional shock wave physics code", *Int. J. Impact Engng.*, Vol. 10, 351 – 360.

Youngs, D. 1987. "An Interface Tracking Method for a 3D Eulerian Hydrodynamics Code", Atomic Weapons Research Establishment Report No. AWRE/44/92/35.